Numerical Algorithms (2006) **41:** 173–196
DOI: 10.1007/s11075-005-9008-0

On memory gradient method with trust region for unconstrained optimization*

Zen-Jun Shi [a,b] and Jie Shen [b]

[a] *College of Operations Research and Management, Qufu Normal University, Rizhao, Shandong 276826, People's Republic of China*
E-mail: zjshi@qrnu.edu.cn, zjshi@lsec.cc.ac.cn
[b] *Department of Computer & Information Science, University of Michigan, Dearborn, MI 48128, USA*
E-mail: shen@umich.edu

Accepted 29 August 2005
Communicated by Xiaojun Chen

In this paper we present a new memory gradient method with trust region for unconstrained optimization problems. The method combines line search method and trust region method to generate new iterative points at each iteration and therefore has both advantages of line search method and trust region method. It sufficiently uses the previous multi-step iterative information at each iteration and avoids the storage and computation of matrices associated with the Hessian of objective functions, so that it is suitable to solve large scale optimization problems. We also design an implementable version of this method and analyze its global convergence under weak conditions. This idea enables us to design some quick convergent, effective, and robust algorithms since it uses more information from previous iterative steps. Numerical experiments show that the new method is effective, stable and robust in practical computation, compared with other similar methods.

Keywords: unconstrained optimization, memory gradient method, global convergence

AMS subject classification: 90C30, 49M37, 65K05

1. Introduction

The line search method for an unconstrained optimization problem

$$\min f(x), \quad x \in R^n, \tag{1}$$

with $f : R^n \to R^1$ being a continuously differentiable function usually takes the form

$$x_{k+1} = x_k + \alpha_k d_k, \; k = 1, 2, \ldots, \tag{2}$$

where d_k is a descent direction, and α_k is a step size. The method is useful and powerful in solving large scale unconstrained optimization problems [14, 15, 17].

* The work was supported in part by NSF DMI-0514900.

Obviously, different choices of d_k and α_k at each iteration will determine different line search methods [22, 23, 28, 30].

If x_k is the current iterate, then we denote $\nabla f(x_k)$ by $g_k, f(x_k)$ by f_k and $f(x^*)$ by f^*, respectively.

Line search method is divided into two stages at each iteration:

(1) Choose a descent search direction d_k;

(2) Choose a step size α_k along the search direction d_k.

If we take $d_k = -g_k$ as a search direction at each iteration, then the corresponding method is called steepest descent method that is a simple line search method and has wide applications in solving large-scale minimization problems (e.g., [19, 20, 30]). However, steepest descent method often yields zigzag phenomena in solving practical problems, which makes the algorithm converge to an optimal solution very slowly, or even fail to converge [14, 15, 17].

Sometimes, choosing step size is also very important for convergence of an algorithm [8, 19–21]. It seems that the step size determines the global convergence and the search direction determines convergence rate in some sense (e.g., [8, 20, 30]).

Generally, the conjugate gradient method is a powerful line search method for solving large scale optimization problems because it avoids, like steepest descent method, the computation and the storage of some matrices associated with the Hessian of objective functions. Conjugate gradient method has the form

$$d_k = \begin{cases} -g_k, & \text{if } k = 1; \\ -g_k + \beta_k d_{k-1}, & \text{if } k \geq 2, \end{cases} \tag{3}$$

where β_k is a parameter that determines the different conjugate gradient methods e.g. [6, 7, 10–13, 24, 25].

Similar method to conjugate gradient method is memory gradient method or super-memory gradient method [3, 16, 26], which also avoids the computation and storage of matrices associated with Newton type method. Therefore, it is also suitable to solve large scale optimization problems. There are other gradient descent methods (e.g. [21–23, 28]) that have good performance in computation.

If we take $d_k = -H_k g_k$ as a search direction at each iteration in the algorithm, where H_k is an $n \times n$ matrix approximating $[\nabla^2 f(x_k)]^{-1}$, then we call the corresponding method Newton-like method [14, 15, 17] such as Newton Method, quasi-Newton method, variable metric method, etc. However, the Newton-like method needs to store and compute matrix H_k at each iteration and thus adds the cost of storage and computation. Accordingly, the Newton-like method is not suitable to solve large scale optimization problems in many situations.

Constructing line search methods not only needs to choose a good search direction d_k but also needs to find an appropriate step size α_k at each iteration. Certainly, trust region methods are also effective methods for solving special optimization problems. In trust region methods, no line search rule is needed. In each iteration of trust region methods, we need only to solve a simple sub-problem [5, 14, 15, 17, 27].

In this paper we present a new memory gradient method with trust region for unconstrained optimization problems. The new method combines line search method and trust region method to generate new iterative points at each iteration. It sufficiently uses the previous multi-step iterative information at each iteration and avoids the storage and computation of matrices associated with the Hessian of objective functions, so that it is suitable to solve large scale optimization problems. We also design an implementable version of this method and analyze its global convergence under weak conditions. This idea enables us to design some quick convergent, effective, and robust algorithms since it uses more information from previous iterative steps. Numerical experiments show that the new method is effective, stable and robust in practical computation.

The rest of this paper is organized as follows. The next section describes some preliminary results. Section 3 describes the idea of memory gradient method with trust region and gives some simple properties. In section 4 we propose a convergent version of the new method and analyze its global convergence under mild conditions. We study the convergence rate of the new method under weak conditions in section 5. Numerical experiments and comparisons are given in section 6.

2. Preliminary

In this section we will recall the line search method and the trust region method and discuss their relationship.

2.1. *Line search method*

Line search methods are important methods for solving unconstrained optimization problems. As described in section 1, in each iteration, line search method needs to choose a search direction d_k first, then it needs to choose a step size α_k along the search direction.

Generally, we choose the search direction d_k to satisfy the descent condition, i.e., there exists a constant $\overline{\alpha}_k > 0$ such that

$$f(x_k + \alpha d_k) < f(x_k), \quad \forall \alpha \in (0, \overline{\alpha}_k). \tag{4}$$

In fact, if $g_k^T d_k < 0$ then d_k must be a descent direction of $f(x)$ at the point x_k.

Furthermore, we may choose d_k to satisfy the following so-called sufficient descent condition

$$g_k^T d_k \leq -\eta_0 \|g_k\|^2, \tag{5}$$

where $\eta_0 > 0$ is a constant.

Finally, we can choose d_k to satisfy the following condition

$$-\frac{g_k^T d_k}{\|g_k\| \cdot \|d_k\|} \geq \eta_0, \tag{6}$$

where η_0 is a positive constant and $\eta_0 \leq 1$. This condition is sometimes called angle property in which the angle of $-g_k$ and d_k needs to be less than $\pi/2$.

Definition 2.1. ([2]). Let $\{x_k\}$ be a sequence generated by a gradient method $x_{k+1} = x_k + \alpha_k d_k$ $(k = 1, 2, \ldots,)$. We say that the sequence $\{d_k\}$ is uniformly gradient related to $\{x_k\}$ if for every convergent subsequence $\{x_k\}_K$, for which

$$\lim_{k \in K, k \to \infty} g_k \neq 0,$$

there holds

$$0 < \liminf_{k \in K, k \to \infty} |g_k^T d_k|, \quad \limsup_{k \in K, k \to \infty} \|d_k\| < \infty.$$

In words, $\{d_k\}$ is uniformly gradient related if whenever a subsequence $\{g_k\}_K$ tends to a nonzero vector, the corresponding subsequence of directions $\{d_k\}$ is bounded and does not tend to be orthogonal to g_k.

The following line search rules for finding step size are often used in many line search methods.

(a) Exact minimization rule. Here α_k is chosen so that

$$f(x_k + \alpha_k d_k) = \min_{\alpha \geq 0} f(x_k + \alpha d_k).$$

(b) Limited minimization rule. A fixed number $s > 0$ is selected and α_k is chosen so that

$$f(x_k + \alpha_k d_k) = \min_{\alpha \in [0,s]} f(x_k + \alpha d_k).$$

(c) Armijo rule. A fixed number $s > 0$ is selected and α_k is the largest α in $\{s, s/2, s/2^2, \ldots\}$ such that

$$f_k - f(x_k + \alpha d_k) \geq -\mu_1 \alpha g_k^T d_k,$$

where $\mu_1 \in (0, 1/2)$.

(d) Goldstein rule. A fixed scalar $\mu_1 \in (0, \frac{1}{2})$ is selected, $\mu_2 \in (\mu_1, 1)$, and α_k is chosen to satisfy

$$\mu_1 \leq \frac{f(x_k + \alpha_k d_k) - f_k}{\alpha_k g_k^T d_k} \leq \mu_2.$$

It is possible to show that if f is bounded below then there exists an interval of step-sizes α_k for which the relation above is satisfied.

(e) Strong Wolfe search rule. The step size α_k satisfies simultaneously

$$f(x_k + \alpha_k d_k) - f_k \leq \alpha_k \mu_1 g_k^T d_k, \tag{7}$$

and

$$|g(x_k + \alpha_k d_k)^T d_k| \leq \mu_2 |g_k^T d_k|, \tag{8}$$

where $0 < \mu_1 < \frac{1}{2}$ and $\mu_1 < \mu_2 < 1$.

It can be easily proved that if $f(x)$ is bounded below, then there also exists an interval of step-sizes α_k for which equations (7) and (8) hold.

We can see that (a) and (b) are exact line search rules. It may be difficult to implement in many situations, so that we often use inexact line search rules in many algorithms.

Lemma 2.1. ([2]). Let $\{x_k\}$ be a sequence generated by a gradient method and assume that $\{d_k\}$ is uniformly gradient related and α_k is chosen by the line search rules (a), (b), (c), (d) or (e). Then every limit point of $\{x_k\}$ is a critical point x^*, i.e., $g(x^*) = 0$.

It is obvious that many line search methods need only the current iterative information explicitly to generate the next iterative points. The previous iterative information is ignored and leads to a waste of information. Thus we should sufficiently use the previous iterative information to generate new iterative points. The idea enables us to design some powerful, robust, effective, and implementable methods for solving large scale unconstrained optimization problems.

2.2. *Trust region method*

In trust region method, we must solve the subproblem at each iteration

$$\min_{p\in R^n} m_k(p) = f_k + g_k^T p + \frac{1}{2} p^T B_k p, \quad s.t. \|p\| \leq \Delta_k, \tag{9}$$

where $\Delta_k > 0$ is a trust region radius and B_k is a symmetric matrix that approximates to $\nabla^2 f(x_k)$.

We define

$$\rho_k = \frac{f_k - f(x_k + p_k)}{m_k(0) - m_k(p_k)} \tag{10}$$

at the k-th iteration, where p_k is a solution of equation (9), the numerator is called actual reduction and the denominator is called predicted reduction.

Algorithm 2.1. (trust region)

```
Given $\overline{\Delta} > 0$, $x_0 \in R^n$, $\Delta_0 \in (0, \overline{\Delta})$, and $\eta \in [0, \frac{1}{4})$;
For $k = 0, 1, 2, \ldots$
  Obtain $p_k$ by(or approximately) solving equation (9);
  Evaluate $\rho_k$ from equation (10);
  if $\rho_k < \frac{1}{4}$
      $\Delta_{k+1} = \frac{1}{4} \|p_k\|$
  else
      if $\rho_k > \frac{3}{4}$ and $\|p_k\| = \Delta_k$
          $\Delta_{k+1} = \min(2\Delta_k, \overline{\Delta})$
      else
          $\Delta_{k+1} = \Delta_k$;
      if $\rho_k > \eta$
          $x_{k+1} = x_k + p_k$
      else
          $x_{k+1} = x_k$;
end(for).
```

Sometimes, we need not to solve equation (9) exactly, but to find p_k satisfying

$$m_k(0) - m_k(p_k) \geq c_1 \|g_k\| \min\left(\Delta_k, \frac{\|g_k\|}{\|B_k\|}\right), \tag{11}$$

and

$$\|p_k\| \leq \gamma\Delta_k, \tag{12}$$

for $\gamma \geq 1$ and $c_1 \in (0, 1]$.

Indeed, the exact solution p_k^* of equation (9) satisfies equations (11) and (12) [17].

Lemma (2.2). [17]. Let $\eta = 0$ in Algorithm 2.1. Suppose that $\|B_k\| \leq \beta$ for some constant β, that f is continuously differentiable and bounded below on the level set

$$L_0 = \{x \in R^n | f(x) \leq f(x_0)\},$$

and that all approximate solutions of equation (9) satisfy the inequalities (11) and (12), for some positive constants c_1 and γ. We then have

$$\liminf_{k\to\infty} \|g_k\| = 0. \tag{13}$$

Lemma 2.3. ([17]). Let $\eta \in (0, \frac{1}{4})$ in Algorithm 2.1. Suppose that $\|B_k\| \leq \beta$ for some constant β, that f is Lipschitz continuously differentiable and bounded below on the level set L_0 and that all approximate solutions of equation (9) satisfy the inequalities (11) and (12), for some positive constants c_1 and γ. We then have

$$\lim_{k\to\infty} \|g_k\| = 0. \tag{14}$$

In trust region algorithm, we need at least to solve inequalities (11) and (12) at each iteration. Moreover, we must store the matrix B_k in implementing the trust region algorithm.

In order to solve large scale unconstrained optimization problems, we must consider the amount of storage and computation in algorithm design. We expect the algorithm to have less amount of storage and computation. Moreover, we hope that the new algorithm is effective, robust, and implementable in practice. Line search methods are one-dimensional search methods while trust region methods are n-dimensional search methods. Although line search methods and trust region methods are very effective in many situations, the two classes of methods have the same disadvantage of only using current iterative information at each iteration. In fact, we expect to use more previous iterative information to generate new iterative points. Accordingly, we propose a new memory gradient method with trust region to overcome the defect of line search methods and trust region methods.

3. Memory gradient method

We assume that the iterative points $x_1, x_2, \ldots, x_m$ have been determined and let $m > 2$ be an integer. We construct a new iterative point x_{k+1} from the previous m-step iterative information of $x_{k-m+1}, \ldots, x_k$, $k = m, m+1, \ldots$. For example, we first constitute a parallel subspace

$$A_k = x_k + span\{-g_{k-m+1}, \ldots, -g_k\},$$

then minimize $f(x)$ over A_k to obtain the minimizer x_{k+1}, i.e.,

$$x_{k+1} = \arg \min_{x \in A_k} f(x), \tag{15}$$

where $span\{-g_{k-m+1}, \ldots, -g_k\}$ denotes a linear subspace spanned by the m vectors $-g_{k-m+1}, \ldots, -g_k$. We call this technique m-step memory gradient method. This idea can be seen in many literatures (e.g., [4, 18, 31]).

We evoke the conjugate gradient method. If x_k is the current iterative point (suppose that $k > 1$), then the next iterative point x_{k+1} is defined on the parallel subspace $A_2 = x_k + span\{-g_k, d_{k-1}\}$. We may call conjugate gradient method two-step memory gradient method (e.g., [3, 7, 13, 16, 24, 25]).

However, solving equation (15) is difficult to realize in practical computation. Thus we must find some inexact multi-dimensional search rules. Firstly, like inexact line search rules, we may design some inexact multi-dimensional search rules. Secondly, we use the idea of trust region methods in multi-dimensional search methods. Define a minimization subproblem

$$\min_{p \in R^n} m_k(p) = f_k + g_k^T p + \frac{1}{2} p^T B_k p, \quad s.t. \|p\| \leq \Delta_k, p \in A_k - x_k, \tag{16}$$

where $\Delta_k > 0$ is a trust region radius,

$$A_k - x_k = span\{-g_{k-m+1}, \ldots, -g_k\}.$$

Obviously, it is an m-dimensional minimization problem. Also, we need not to solve equation (16) exactly, but to find p_k satisfying

$$m_k(0) - m_k(p_k) \geq c_1 \|g_k\| \min\left(\Delta_k, \frac{\|g_k\|}{\|B_k\|}\right), \tag{17}$$

and

$$\|p_k\| \leq \gamma\Delta_k, p_k \in A_k - x_k, \tag{18}$$

for some constants $\gamma \geq 1$ and $c_1 \in (0, 1]$.

As you can see, if $n >> m$ then equations (17) and (18) are easier to solve than equations (11) and (12) because equations (17) and (18) are actually inequalities of m variables, while equation (11) and (12) are inequalities of n variables.

Now we should devise an algorithmic model

Algorithm model

```
Given $\overline{\Delta} > 0$, $x_0 \in R^n$, $\Delta_0 \in (0, \overline{\Delta})$, and $\eta \in [0, \frac{1}{4})$;
For $k = 0, 1, 2, \ldots$
   Obtain $p_k$ by (or approximately) solving equation (16);
   Evaluate $\rho_k$ from equation (10);
   if $\rho_k < \frac{1}{4}$
       $\Delta_{k+1} = \frac{1}{4}\|p_k\|$
   else
       if $\rho_k > \frac{3}{4}$ and $\|p_k\| = \Delta_k$
          $\Delta_{k+1} = \min(2\Delta_k, \overline{\Delta})$
       else
          $\Delta_{k+1} = \Delta_k$;
   if $\rho_k > \eta$
       $x_{k+1} = x_k + p_k$
   else
       $x_{k+1} = x_k$;
end(for).
```

Lemma 3.1. Suppose that $\|B_k\| \leq \beta$ for some constant β, that f is continuously differentiable and bounded below on the level set L_0. Then there exists p_k satisfying equations (17) and (18) and hence satisfying equations (11) and (12).

Proof. At first, letting

$$V_k = (-g_{k-m+1}, \ldots, -g_k) \in R^{n\times m},\ y = (0, \ldots, 0, y_m)^T \in R^m,\ y_m \in R^1,$$

we have $p = V_k y = -y_m g_k \in A_k - x_k$. By substituting this p into equation (16), we obtain

$$\min_{y_m \in R^1} m_k(-y_m g_k) = f_k - y_m \|g_k\|^2 + \frac{1}{2} y_m^2 g_k^T B_k g_k, \quad s.t.\ \|y_m g_k\| \le \Delta_k.$$

Solve this subproblem and obtain a minimizer y'_m. In fact, in the case of $g_k^T B_k g_k \le 0$ we have $y'_m = \Delta_k/\|g_k\|$; in the case of $g_k^T B_k g_k > 0$, if $\|g_k\|^2/g_k^T B_k g_k \le \Delta_k/\|g_k\|$ then $y'_m = \|g_k\|^2/g_k^T B_k g_k$ else $y'_m = \Delta_k/\|g_k\|$. Therefore, in the case of $g_k^T B_k g_k \le 0$ we have

$$\begin{aligned} m_k(0) - m_k(-y'_m g_k) &= y'_m \|g_k\|^2 - \frac{1}{2} y'^2_m g_k^T B_k g_k \\ &\ge \|g_k\| \Delta_k \\ &\ge \|g_k\| \min\left(\Delta_k, \frac{\|g_k\|}{\|B_k\|}\right). \end{aligned}$$

In the case of $g_k^T B_k g_k > 0$, if $\|g_k\|^2/g_k^T B_k g_k \le \Delta_k/\|g_k\|$ then we have

$$\begin{aligned} m_k(0) - m_k(-y'_m g_k) &= y'_m \|g_k\|^2 - \frac{1}{2} y'^2_m g_k^T B_k g_k \\ &= \|g_k\|^2 \frac{\|g_k\|^2}{g_k^T B_k g_k} - \frac{1}{2} \frac{\|g_k\|^4}{(g_k^T B_k g_k)^2} g_k^T B_k g_k \\ &= \frac{1}{2} \frac{\|g_k\|^4}{g_k^T B_k g_k} \\ &\ge \frac{1}{2} \frac{\|g_k\|^2}{\|B_k\|} \\ &\ge \frac{1}{2} \|g_k\| \min\left(\Delta_k, \frac{\|g_k\|}{\|B_k\|}\right), \end{aligned}$$

else, since $\|g_k\|^2/g_k^T B_k g_k > \Delta_k/\|g_k\|$, we have $g_k^T B_k g_k/\|g_k\|^2 < \|g_k\|/\Delta_k$, and thus

$$\begin{aligned} m_k(0) - m_k(-y'_m g_k) &= \|g_k\| \Delta_k - \frac{1}{2} \left(\frac{\Delta_k}{\|g_k\|}\right) g_k^T B_k g_k \\ &= \Delta_k \|g_k\| - \frac{1}{2} \Delta_k^2 \cdot \frac{g_k^T B_k g_k}{\|g_k\|^2} \\ &\ge \Delta_k \|g_k\| - \frac{1}{2} \Delta_k \|g_k\| \\ &= \frac{1}{2} \Delta_k \|g_k\| \\ &\ge \frac{1}{2} \|g_k\| \min\left(\Delta_k, \frac{\|g_k\|}{\|B_k\|}\right). \end{aligned}$$

By letting $c_1 = 1/2$, it follows that

$$m_k(0) - m_k(-y'_m g_k) \geq c_1 \|g_k\| \min\left(\Delta_k, \frac{\|g_k\|}{\|B_k\|}\right).$$

This shows that $p_k = -y'_m g_k \in A_k - x_k$ satisfies equations (17) and (18) and hence satisfies equations (11) and (12) for $c_1 = 1/2$. If p_k is a solution to the subproblem equation (16) then we have

$$m_k(0) - m_k(p_k) \geq m_k(0) - m_k(-y'_m g_k).$$

The proof is completed. □

We can obtain similar results as in Lemmas 2.2 and 2.3 by using Lemma 3.1 and Lemmas 2.2 and 2.3.

Theorem 3.1. Let $\eta = 0$ in the Algorithm model. Suppose that $\|B_k\| \leq \beta$ for some constant β, that f is continuously differentiable and bounded below on the level set L_0 and that all approximate solutions of equation (16) satisfy the inequalities (17) and (18), for some positive constants c_1 and γ. We then have

$$\liminf_{k\to\infty} \|g_k\| = 0.$$

Theorem 3.2. Let $\eta \in (0, \frac{1}{4})$ in the Algorithm model. Suppose that $\|B_k\| \leq \beta$ for some constant β, that f is Lipschitz continuously differentiable and bounded below on the level set L_0 and that all approximate solutions of equation (16) satisfy the inequalities (17) and (18), for some positive constants c_1 and γ. We then have

$$\lim_{k\to\infty} \|g_k\| = 0.$$

Theorems 3.1 and 3.2 are essentially similar to Lemmas 2.2 and 2.3. However, solving equations (17) and (18) is also difficult. From the view of computation and storage, line search methods seem to be easier to implement than trust region methods. Therefore line search methods are more suitable to solve large scale optimization problems than trust region methods.

We expect to combine line search methods and trust region methods to construct a new memory gradient method with trust region. We hope that the new algorithm is not only implementable, effective and robust, but also has less costs of storage and computation.

4. A convergent version of the new method

We assume that

(H1). The function $f(x)$ has a lower bound on R^n.

(H2). The gradient $g(x)$ is Lipschitz continuous in an open convex set B that contains $L_0 = \{x \in R^n | f(x) \leq f(x_0)\}$, where x_0 is given, i.e., there exists an $L > 0$ such that

$$\|g(x) - g(y)\| \leq L\|x - y\|, \quad \forall x, y \in B. \tag{19}$$

(H3). The gradient $g(x)$ of $f(x)$ is uniformly continuous in an open convex set B that contains L_0.

It is apparent that Assumption ($H2$) implies ($H3$).

Suppose that $m > 0$ is an integer and m_k is defined by

$$m_k = \min(k, m). \tag{20}$$

We describe an implementable memory gradient method with trust region as follows.

Algorithm (A).

Step 0. Given $0 < \mu_1 < \frac{1}{2}$, $\mu_1 < \mu_2 < 1$, a fixed integer $m \geq 2$, $s \in (m-1, \infty)$, $L_1 \geq 0$, and $x_1 \in R^n$, set $k := 1$;

Step 1. If $\|g_k\| = 0$ then stop; else go to Step 2;

Step 2. $x_{k+1} = x_k + \alpha_k d_k(\beta^{(k)}_{k-m_k+1}, \ldots, \beta^{(k)}_k)$ where

$$d_k(\beta^{(k)}_{k-m_k+1}, \ldots, \beta^{(k)}_k) = -\sum_{i=1}^{m_k} \beta^{(k)}_{k-i+1} g_{k-i+1}, \tag{21}$$

and $\{\beta^{(k)}_{k-i+1}\}(i = 1, 2, \ldots, m_k)$ is a solution to the minimization problem (SP):

$$\min\{g_k^T d_k(\beta_{k-m_k+1}, \ldots, \beta_k) + \tfrac{1}{2} L_k \|d_k(\beta_{k-m_k+1}, \ldots, \beta_k)\|^2\},$$
$$s.t. \quad \sum_{i=1}^{m_k} \beta_{k-i+1} \geq s, \ \beta_{k-i+1} \in [0, s_k^i], i = 1, 2, \ldots, m_k \ ,$$

with

$$L_k \geq 0, \ s_k^1 = s, \ s_k^i = \frac{\|g_k\|^2}{\|g_k\|^2 + |g_k^T g_{k-i+1}|}, \ (i = 2, \ldots, m_k),$$

and α_k is chosen by the line search rules (a), (b), (c), (d) or (e);

Step 3. Set $k := k + 1$ and goto Step 1.

For simplicity, we denote $d_k(\beta^{(k)}_{k-m_k+1}, \ldots, \beta^{(k)}_k)$ by d_k throughout the paper and $\|\cdot\|$ denotes the Euclidean norm on R^n.

If $L_k = 0$, we can solve the subproblem (SP) by implementing the following procedure:

$$\beta_k^{(k)} = s; \ \text{if} \ g_k^T g_{k-i+1} \geq 0, \ \text{then} \ \beta^{(k)}_{k-i+1} = s_k^i, \ \text{else} \ \beta^{(k)}_{k-i+1} = 0(i = 2, 3, \ldots, m_k).$$

Generally, for $k \geq 2$, we estimate L_k by Barzilai and Browein's technique [1], that is, L_k is a solution to the minimization problem

$$\min_{L \in R^1} \|L(x_k - x_{k-1}) - (g_k - g_{k-1})\|,$$

and thus

$$L_k = \frac{(x_k - x_{k-1})^T (g_k - g_{k-1})}{\|x_k - x_{k-1}\|^2}.$$

Lemma 4.1. For all $k \geq 1$,

$$g_k^T d_k \leq -(s - m + 1)\|g_k\|^2.$$

Proof. By $\beta_{k-i+1}^{(k)} \leq s_k^i,\ (i = 1, \ldots, m_k)$, we have

$$\begin{aligned}
-g_k^T d_k &= \sum_{i=1}^{m_k} \beta_{k-i+1}^{(k)} g_k^T g_{k-i+1} \\
&= \beta_k^{(k)} \|g_k\|^2 + \sum_{i=2}^{m_k} \beta_{k-i+1}^{(k)} g_k^T g_{k-i+1} \\
&\geq \left(s - \sum_{i=2}^{m_k} \beta_{k-i+1}^{(k)} \right) \|g_k\|^2 + \sum_{k=2}^{m_k} \beta_{k-i+1}^{(k)} g_k^T g_{k-i+1} \\
&= s\|g_k\|^2 - \sum_{i=2}^{m_k} \beta_{k-i+1}^{(k)} [\|g_k\|^2 - g_k^T g_{k-i+1}] \\
&\geq s\|g_k\|^2 - \sum_{k=2}^{m_k} s_k^i [\|g_k\|^2 + |g_k^T g_{k-i+1}|] \\
&= s\|g_k\|^2 - (m_k - 1)\|g_k\|^2 \\
&\geq (s - m + 1)\|g_k\|^2.
\end{aligned}$$

This completes the proof. □

Lemma 4.2. For all $k \geq 1$,

$$\|d_k\| \leq 2s\delta_k,$$

where

$$\delta_k = \max_{1 \leq i \leq k} \{\|g_i\|\}.$$

Proof. The conclusion is obvious. The proof is omitted.

Lemma 4.2 shows that d_k is in a trust region, and we can call this method a memory gradient method with trust region. Thereby, the memory gradient method with trust region has both the advantage of simple structure of line search methods and the advantage of strong convergence of trust region methods in some sense. Other similar memory gradient methods have no such a property [3, 4, 26].

Lemma 4.3. If $(H1)$ and $(H3)$ hold, Algorithm (A) with the line search rules (c), (d) or (e) generates an infinite sequence $\{x_k\}$, then the sequence $\{\|g_k\|\}$ and $\{\|d_k\|\}$ are bounded for all $k \geq 1$.

Proof. From Lemma 4.2, it is sufficient to prove that $\{\|g_k\|\}$ has a bound. We use reduction to absurdity to prove the conclusion. Suppose that $\{\|g_k\|\}$ has no bound, then

$$\lim_{k\to\infty} \delta_k = \infty.$$

It is shown that there exists an infinite subset $N \subseteq \{m, m+1, \ldots\}$ such that

$$\delta_k = \|g_k\| \to \infty, \forall k \in N. \tag{22}$$

By $(H1)$, line search rules (c), (d) and (e), we can obtain that $\{f_k\}$ is a decreasing sequence and has a lower bound, thus $\{f_k\}$ is a convergent sequence. Therefore

$$\sum_{k=m}^{\infty} \alpha_k(-g_k^T d_k) < \infty.$$

By Lemma 4.1, we have

$$\sum_{k=m}^{\infty} \alpha_k \|g_k\|^2 < \infty. \tag{23}$$

Since $N \subseteq \{m, m+1, \ldots\}$, by equation (23), we have

$$\sum_{k\in N} \alpha_k \|g_k\|^2 \leq \sum_{k=m}^{\infty} \alpha_k \|g_k\|^2 < \infty. \tag{24}$$

By Lemma 4.2 and equation (22), we obtain

$$\|d_k\| \leq 2s\delta_k = 2s\|g_k\|, \forall k \in N. \tag{25}$$

From equations (24) and (25), it holds that

$$\sum_{k\in N} \alpha_k \|d_k\|^2 < \infty, \tag{26}$$

thus

$$\alpha_k \|d_k\|^2 \to 0 (k \in N, k \to \infty). \tag{27}$$

By Lemma 4.1 and Cauchy–Schwartz inequality, we have $\|g_k\| \cdot \|d_k\| \geq -g_k^T d_k \geq (s-m+1)\|g_k\|^2$, and thus $\|d_k\| \geq -g_k^T d_k \geq (s-m+1)\|g_k\|$. It follows from equation (22) that

$$\|d_k\| \to \infty (k \in N, k \to \infty).$$

By equation (27), we have

$$\alpha_k \|d_k\| \to 0 (k \in N, k \to \infty). \tag{28}$$

For Armijo line search rule (c), let $N_1 = \{k|\ \alpha_k = s\}$, $N_2 = \{k|\ \alpha_k < s\}$, by equation (23) we have

$$\sum_{k\in N_1} \alpha_k \|g_k\|^2 + \sum_{k\in N_2} \alpha_k \|g_k\|^2 < \infty,$$

and thus, if N_1 is an infinite subset, then

$$\sum_{k\in N_1} \alpha_k \|g_k\|^2 = s \sum_{k\in N_1} \|g_k\|^2 < \infty,$$

therefore $\|g_k\| \to 0(k \in N_1, k \to \infty)$. This shows that $N \subseteq N_2$ and $2\alpha_k \le s, \forall k \in N$. From Armijo line search rule, it must hold that

$$f_k - f(x_k + 2\alpha_k d_k) < -2\mu_2 \alpha_k g_k^T d_k, \forall k \in N. \tag{29}$$

By using the mean value theorem, there exists $\theta_k \in [0, 1]$ such that

$$f_k - f(x_k + 2\alpha_k d_k) = -2\alpha_k g(x_k + 2\alpha_k \theta_k d_k)^T d_k.$$

Noting equation (29), we have

$$g(x_k + 2\alpha_k \theta_k d_k)^T d_k > \mu_2 g_k^T d_k, \forall k \in N. \tag{30}$$

By Lemma 4.1, equations (25) and (30) we have

$$\begin{aligned}(s - m + 1)(1 - \mu_2)\|g_k\|^2 &\le (1 - \mu_2)(-g_k^T d_k)\\ &\le (g(x_k + 2\alpha_k \theta_k d_k) - g_k)^T d_k\\ &\le \|d_k\| \cdot \|g(x_k + 2\alpha_k \theta_k d_k) - g_k\|\\ &\le 2s\|g_k\| \cdot \|g(x_k + 2\alpha_k \theta_k d_k) - g_k\|, \forall k \in N,\end{aligned}$$

thus

$$(s - m + 1)(1 - \mu_2)\|g_k\| \le 2s\|g(x_k + 2\alpha_k \theta_k d_k) - g_k\|, \forall k \in N.$$

By $(H3)$ and equation (28), we have

$$\|g_k\| \to 0(k \in N, k \to \infty).$$

This contradicts equation (22), therefore $\{\|g_k\|\}$ has a bound.

For Goldstein line search rule (d), by the mean value theorem, there exists $\theta_k \in [0, 1]$ such that

$$-\alpha_k g(x_k + \alpha_k \theta_k d_k)^T d_k = f_k - f_{k+1} \le -\mu_2 g_k^T d_k,$$

thus

$$g(x_k + \alpha_k \theta_k d_k)^T d_k \ge \mu_2 g_k^T d_k. \tag{31}$$

By equations (31) and (25), Lemmas 4.1 and 4.2, and Cauchy-Schwartz inequality, we have

$$\begin{aligned}(s-m+1)(1-\mu_2)\|g_k\|^2 &\le (1-\mu_2)(-g_k^T d_k)\\ &\le (g(x_k+\alpha_k\theta_k d_k)-g_k)^T d_k\\ &\le \|g(x_k+\alpha_k\theta_k d_k)-g_k)\|\cdot\|d_k\|\\ &\le 2s\|g(x_k+\alpha_k\theta_k d_k)-g_k\|\cdot\|g_k\|, \forall k\in N,\end{aligned}$$

thus

$$(s-m+1)(1-\mu_2)\|g_k\| \le 2s\|g(x_k+\alpha_k\theta_k d_k)-g_k\|.$$

Noting $(H3)$, equation (28), we obtain that

$$\|g_k\|\to 0 (k\in N, k\to\infty).$$

This contradicts equation (22), therefore the boundedness of $\{\|g_k\|\}$ is proved.

For Wolfe line search rule (e), we have

$$g(x_k+\alpha_k d_k)^T d_k \ge \mu_2 g_k^T d_k, \tag{32}$$

thus, by Lemmas 4.1 and 4.2, equation (25) and Cauchy–Schwartz inequality, we obtain

$$\begin{aligned}(s-m+1)(1-\mu_2)\|g_k\|^2 &\le (1-\mu_2)(-g_k^T d_k)\\ &\le (g_{k+1}-g_k)^T d_k\\ &\le \|g_{k+1}-g_k\|\cdot\|d_k\|\\ &\le 2s\|g_k\|\cdot\|g_{k+1}-g_k\|, \forall k\in N.\end{aligned}$$

Therefore

$$(s-m+1)(1-\mu_2)\|g_k\| \le 2s\|g_{k+1}-g_k\|.$$

Noting $(H3)$ and equation (28) we obtain

$$\|g_k\|\to 0 (k\in N, k\to\infty).$$

This contradicts equation (22), therefore the boundedness of $\{\|g_k\|\}$ is proved.

In summary, we obtain that $\{\|g_k\|\}$ has a bound. By Lemma 4.2, $\{\|d_k\|\}$ has also a bound. The proof is complete. □

Theorem 4.1. Assume that $(H1)$ and $(H3)$ hold, Algorithm (A) with the line search rules (c), (d) or (e) generates an infinite sequence $\{x_k\}$, then

$$\lim_{k\to\infty}\|g_k\| = 0. \tag{33}$$

Proof. By Lemmas 4.1 and 4.3, it follows that the search direction sequence $\{d_k\}$ is uniformly gradient related to $\{x_k\}$. By Lemma 2.1, the conclusion is proved. □

Corollary 4.1. Assume that $(H1)$ and $(H2)$ hold, Algorithm (A) with the line search rules (c), (d) or (e) generates an infinite sequence $\{x_k\}$, then equation (33) holds.

Proof. Because Assumption $(H2)$ implies $(H3)$, in this case the conditions of Theorem 4.1 hold, thus equation (33) holds. □

Lemma 4.2. If $(H1)$ and $(H2)$ hold, Algorithm (A) with the line search rules (a) and (b) generates an infinite sequence $\{x_k\}$, then the sequences $\{\|g_k\|\}$ and $\{\|d_k\|\}$ are bounded for all $k \geq 1$.

Proof. By the mean value theorem and $(H2)$ we have

$$\begin{aligned} f(x_k + \alpha d_k) &= f_k + \alpha g_k^T d_k + \alpha \int_0^1 [g(x_k + t\alpha_k d_k) - g_k]^T d_k dt \\ &\leq f_k + \alpha g_k^T d_k + \alpha \int_0^1 \|g(x_k + t\alpha d_k) - g_k\| \cdot \|d_k\|^2 dt \\ &\leq f_k + \alpha g_k^T d_k + \tfrac{1}{2}\alpha^2 L \|d_k\|^2. \end{aligned}$$

By Lemma 4.1 and letting

$$\alpha = -\frac{g_k^T d_k}{L\|d_k\|^2}$$

in the above inequality we have

$$\begin{aligned} f(x_k + \alpha_k d_k) - f_k &\leq f(x_k + \alpha d_k) \\ &\leq -\frac{1}{2L} \cdot \left(\frac{g_k^T d_k}{\|d_k\|} \right)^2 \\ &\leq -\frac{(s-m+1)^2}{2L} \cdot \frac{\|g_k\|^4}{\|d_k\|^2}, \end{aligned}$$

where α_k is defined by line search rules (a) or (b). By $(H1)$ and Lemma 4.2 we have

$$\sum_{k=1}^{\infty} \frac{\|g_k\|^4}{2s\delta_k^2} \leq \sum_{k=1}^{\infty} \frac{\|g_k\|^4}{\|d_k\|^2} < \infty.$$

If $\{\|g_k\|\}$ has no bound then there exists an infinite subset $N \subseteq \{m, m+1, \ldots\}$ such that equation (22) holds. Thus

$$\frac{1}{2s} \sum_{k \in N} \|g_k\|^2 \leq \frac{1}{2s} \sum_{k=1}^{\infty} \frac{\|g_k\|^4}{\delta_k^2} < \infty$$

which contradicts equation (22). The proof is complete. □

Theorem 4.2. If $(H1)$ and $(H2)$ hold, Algorithm (A) with the line search rules (a) and (b) generates an infinite sequence $\{x_k\}$, then

$$\lim_{k \to \infty} \|g_k\| = 0.$$

Proof. By Lemmas 4.1 and 4.3, it follows that the search direction sequence $\{d_k\}$ is uniformly gradient related to $\{x_k\}$. By Lemma 2.1, the conclusion is proved.

5. Linear convergence rate

Lemma 5.1. Assumption ($H2$) holds, Algorithm (A) generates an infinite sequence $\{x_k\}$, then either

$$f_k - f_{k+1} \geq \eta_0 \|g_k\|^2, (k = 1, 2, \ldots), \tag{34}$$

or there exists an infinite subset $N \subseteq \{1, 2, ...\}$ such that

$$\lim_{k \in N, k \to \infty} \left\{ \frac{\|g_k\|^2}{\delta_k^2} \right\} = 0, \tag{35}$$

where η_0 is a constant, and δ_k is defined in Lemma 4.2.
Proof. If the step size α_k satisfies

$$\eta_1 = \inf_{\forall k} \{\alpha_k\} > 0, \tag{36}$$

then by line search rules (d) and (e), and Lemma 4.1, we have

$$\begin{aligned} f_k - f_{k+1} &\geq -\mu_1 \alpha_k g_k^T d_k \\ &\geq \mu_1 \eta_1 (s - m + 1) \|g_k\|^2 \\ &= \eta_0 \|g_k\|^2, \end{aligned}$$

where $\eta_0 = \mu_1 \eta_1 (s - m + 1)$; by line search rule (c) and Lemma 4.1, we also have

$$f_k - f_{k+1} \geq \eta_0 \|g_k\|^2,$$

where $\eta_0 = \mu_1 \eta_1 (s - m + 1)$. This shows that equation (34) holds.

For exact minimization rule (a) and limited minimization rule (b), suppose that α_k^* is the step size defined by (a) or (b) and α_k is the step size defined by (c). Then

$$f_k - f(x_k + \alpha_k^* d_k) \geq f_k - f_{k+1} \geq \eta_0 \|g_k\|^2,$$

which shows that equation (34) also holds for line search rules (a) and (b).

If equation (36) doesn't hold then there exists an infinite subset $N \subset \{m, m+1, \ldots\}$ such that

$$\lim_{k \in N, k \to \infty} \{\alpha_k\} = 0. \tag{37}$$

For line search rules (d) and (e), by equations (31) and (32), Lemma 4.1 and ($H2$), we have

$$\begin{aligned}\alpha_k &\geq \frac{\|g(x_k+\alpha_k\theta_k d_k)-g_k\|\cdot\|d_k\|}{L\|d_k\|^2}\\ &\geq \frac{(g(x_k+\alpha_k\theta_k d_k)-g_k)^T d_k}{L\|d_k\|^2}\\ &\geq -\frac{(1-\mu_2)g_k^T d_k}{L\|d_k\|^2}\\ &\geq \frac{(1-\mu_2)(s-m+1)\|g_k\|^2}{L\delta_k^2}.\end{aligned}$$

From equation (37), it follows that equation (35) holds.

For Armijo line search rule (c), by equation (37), it holds that $\alpha_k < s, \forall k \in N$, thus equation (30) holds. Also by Lemma 4.1, ($H2$) and equation (37), we have

$$\alpha_k \geq \frac{\|g(x_k+2\alpha_k\theta_k d_k)-g_k\|\cdot\|d_k\|}{2L\|d_k\|^2} \geq \frac{(1-\mu_2)(s-m+1)\|g_k\|^2}{2L\delta_k^2},$$

which shows that equation (35) holds.

For exact minimization rules (a) and (b), if equation (37) holds, then we must have $g_{k+1}^T d_k = 0, \forall k \in N$. Therefore,

$$\alpha_k \geq \frac{\|g_{k+1}-g_k\|\cdot\|d_k\|}{L\|d_k\|^2} \geq \frac{(s-m+1)\|g_k\|^2}{L\delta_k^2},$$

which shows that equation (35) holds. □

Assumption. ($H4$). $f(x)$ is twice continuously differentiable, Algorithm (A) generates an infinite sequence $\{x_k\}$ converging to $x*$, and suppose that there exist $M' > m' > 0$ and a neighboring region $N_\delta(x*) = \{x \in R^n | \|x - x*\| < \delta\}$ of $x*$ such that

$$m'\|y\|^2 \leq y^T\nabla^2 f(x)y \leq M'\|y\|^2, \forall x \in N_\delta(x^*), y \in R^n. \tag{38}$$

Lemma 5.2. [15]. If Assumption ($H4$) holds, then f has a unique minimizer x^* in $N_\delta(x^*)$, g is a Lipschitz continuous function on $N_\delta(x^*)$, and

$$m'\|x-x*\|^2 \leq f(x) - f(x^*) \leq \frac{1}{2}M'\|x-x*\|^2,$$

$$m'\|x-x*\| \leq \|g(x)\| \leq M'\|x-x*\|.$$

Theorem 5.1. If Assumption (*H*4) holds, Algorithm (A) generates an infinite sequence $\{x_k\}$ converging to x^*, then either

$$\limsup_{k\to\infty} \|x_k - x^*\|^{\frac{1}{k}} < 1, \tag{39}$$

or there exists an infinite subset $N \subseteq \{1, 2, \ldots\}$ and $i_k \in \{1, 2, \ldots, m\}$ such that

$$\lim_{k\in N, k\to\infty} \frac{\|x_k - x^*\|}{\|x_{k-i_k+1} - x^*\|} = 0. \tag{40}$$

Proof. If equation (34) holds, then equation (39) is proved in a similar way in [15]. If equation (35) holds, then there must exist $i_k \in \{1, 2, \ldots, m\}$ such that $\delta_k = \|g_{k-i_k+1}\|$, thus

$$\lim_{k\in N, k\to\infty} \frac{\|g_k\|^2}{\|g_{k-i_k+1}\|^2} = 0.$$

By Lemma 5.2, we have that equation (40) holds. This completes the proof. □

6. Numerical experiments

The conjugate gradient method takes the form equation (3) in which

$$\beta_k^{FR} = \frac{\|g_k\|^2}{\|g_{k-1}\|^2}, \beta_k^{PRP} = \frac{g_k^T(g_k - g_{k-1})}{\|g_{k-1}\|^2}, \beta_k^{HS} = \frac{g_k^T(g_k - g_{k-1})}{d_{k-1}^T g_{k-1}}.$$

Its corresponding method is called FR, PRP, HS conjugate gradient method (e.g. [2, 14, 15, 17, 24, 25]), respectively. For non-quadratic objective function in equation (1), we use Wolfe line search rule to choose the step size α_k in steepest descent method, and in FR, PRP, HS methods, etc. The new method in the paper is denoted by NM, the steepest descent method by SM. All these methods have the same property that avoids the overhead and evaluation of second derivative of f, the storage and computation of matrix associated with Newton type methods.

We chose some test problems and conducted some numerical experiments, which show that the new algorithm is useful in practical computation.

Test 1. [12]

$$f(x) = (x_1 + 10x_2)^4 + 5(x_3 - x_4)^4 + (x_2 - 2x_3)^4 + 10(x_1 - x_4)^4,$$
$$x_0 = (2, 2, -2, -2)^T, \quad x^* = (0, 0, 0, 0)^T, \quad f^* = 0.$$

Test 2. [6]

$$f = (1 - x_1)^2 + (1 - x_{10})^2 + \sum_{i=1}^{9}(x_i^2 - x_{i+1})^2,$$
$$x_0 = (-2, \ldots, -2)^T, \quad x^* = (1, \ldots, 1)^T, \quad f^* = 0.$$

Test 3. Extended Powell function [9]

$$f = \sum_{i=1}^{n-3} [(x_i + 10x_{i+1})^2 + 5(x_{i+2} - x_{i+3})^2 + (x_{i+1} - 2x_{i+2})^4 + 10(x_i - x_{i+3})^4],$$
$$x_0 = (3, -1, 0, 1, \dots, 3, -1, 0, 1)^T, \quad x^* = (0, 0, \dots, 0)^T, \quad f^* = 0.$$

We used Test 3-9 to denote respectively the cases of n = 100, n = 1000, n = 10000, n = 15000, n = 20000, n = 25000, n = 30000 in Test 3. We take $\rho = 0.87, \ \mu_2 = 0.38, \ \ m = 3, \ \ eps = 10^{-7}$ and use Armijo line search rule in Algorithm (A). The numerical results are reported in table 1. A Computer with Pentium IV/1.2 Gh and C++ programming language are used in implementing the algorithm.

In the numerical experiment, we used Barzilai and Borwein's technique to estimate the parameter L_k in the new method [1]. In fact, L_k is an approximation to the Lipschitz constant of the gradient of $f(x)$. If L is a known priori in Assumption $(H2)$ then we take $L_k \equiv L$, otherwise, we can take

$$L_k = \frac{(g_k - g_{k-1})^T (x_k - x_{k-1})}{\|x_k - x_{k-1}\|^2},$$

or take

$$L_k = \frac{\|g_k - g_{k-1}\|^2}{(x_k - x_{k-1})^T (g_k - g_{k-1})},$$

Table 1
Numerical results.

T	NM	SM	FR	PRP	HS
T1	18, 123	31, 656	80, 1241	26, 523	43, 310
T2	63, 1002	146, 2293	654, 1988	92, 1500	121, 1828
T3	69, 143	204, 528	76, 1691	194, 1205	108, 2034
T4	83, 1326	688, 6506	178, 4157	143, 4734	122, 2748
T5	123, 1641	3,424, 23451	1,273, 6346	781, 3721	918, 7212
T6	168, 2453	524, 6521	428, 5328	594, 6235	612, 5289
T7	193, 3143	1,245, 8362	638, 6279	794, 8261	538, 3483
T8	269, 5133	638, 9280	382, 6392	432, 8462	346, 7392
T9	352, 6442	578, 9210	638, 12763	428, 7621	419, 6893
T10	87, 176	248, 521	152, 317	177, 469	226, 445
T11	63, 126	88, 194	74, 263	68, 185	61, 167
T12	42, 98	66, 127	65, 163	58, 164	61, 137
T13	116, 1432	135, 1987	121, 2138	138, 1817	119, 1734
T14	11, 56	21, 78	27, 82	25, 93	16, 72
T15	8, 33	14, 69	18, 32	12, 47	14, 57
CPU	384 s	860 s	721 s	684 s	727 s

whenever $k \geq 2$. In our numerical experiments, we used the first estimation because of

$$\begin{aligned} L_k &= \frac{(g_k - g_{k-1})^T (x_k - x_{k-1})}{\|x_k - x_{k-1}\|^2} \\ &\leq \frac{\|g_k - g_{k-1}\| \cdot \|x_k - x_{k-1}\|}{\|x_k - x_{k-1}\|^2} \\ &= \frac{\|g_k - g_{k-1}\|}{\|x_k - x_{k-1}\|} \\ &\leq L. \end{aligned}$$

Other test problems, denoted by Tests 10–15, were chosen from the book [29].

Test 10.

$$f(x) = \sum_{i=1}^{20} [(x_1 + x_2 t_i - \exp(t_i))^2 + (x_3 + x_4 \sin(t_i) - \cos(t_i))^2],$$

where $t_i = 0.2i, i = 1, \ldots, 20;\ x_0 = (25, 5, -5, -1)^T$,

$$x^* = (-10.22, 11.91, -0.4580, 0.5803)^T,\ \ f(x^*) = 903.234.$$

Test 11.

$$f(x) = -\left[\sum_{i=1}^{8} x_i^2\right] * \left[\sum_{i=1}^{8} x_i^4\right] + \left[\sum_{i=1}^{8} x_i^3\right]^2,$$

$x_0 = (1, 1, 1, 1, 1, 1, 1, 1)^T,\ x^* = (0.498, 0.499, 0.502, 0.504, 0.503, 0.502, 1, 1)^T$, $f(x^*) = -0.749976$.

Test 12.

$$f(x) = x_1^2 + (x_2 - x_1^2 - 1)^2 + \sum_{i=2}^{30} \left\{ \sum_{j=2}^{9} (j-1) x_j \left(\frac{i-1}{29}\right)^{j-2} - \left[\sum_{j=1}^{9} x_j \left(\frac{i-1}{29}\right)^{j-1}\right]^2 - 1 \right\}^2,$$

$x_0 = 0, x^* = (0.15E - 04, 0.1, -0.0147, -0.146, 1, -2.62, 4.1, -2.14, 1.05)^T$, $f(x^*) = 0.139977E - 05$.

Test 13.

$$f(x) = \sum_{i=1}^{30} \alpha_i(x),$$

$$\alpha_i(x) = 420x_i + (i-15)^3 + \sum_{j=1}^{30} v_{ij}[(\sin(\log(v_{ij})))^5 + (\cos(\log(v_{ij})))^5],$$

where $v_{ij} = \sqrt{x_j^2 + i/j}; x_0 = (x_i; i = 1, \ldots, 30)^T = (-2.8742711\alpha_i(0);$
$i = 1, \ldots, 30)^T.$

$x^* = (7898.84, 6316.09, 4957.31, 3806.63, 2846.71, 2060.11, 1429.46, 937.42, 566.67, 299.90, 119.83, 9.16, -49.40, -73.12, -79.28, -85.16, -108.03, -165.16, -273.82, -451.27, -714.77, -1081.60, -1569, -2194.25, -2974.59, -3927.29, -5069.59, -6418.77, -7992.06, -9806.72)^T, f(x^*) = 0.$

Test 14.

$$f(x) = [-13 + x_1 - 2x_2 + 5x_2^2 - x_2^3]^2 + [-29 + x_1 - 14x_2 + x^2 + x_2^3]^2,$$

where $x_0 = (15, -2)^T,\ x^* = (5, 4),\ f(x^*) = 0.$

Test 15.

$$f(x) = (x_1 - x_2 + x_3)^2 + (-x_1 + x_2 + x_3)^2 + (x_1 + x_2 - x_3)^2,$$

where $x_0 = (100, -1, 2.5)^T,\ x^* = (0, 0, 0)^T,\ f(x^*) = 0.$

With each pair of numbers in table 1, the first number denotes the number of iterations and the second number denotes function evaluations. **CPU** denotes the total CPU time for solving all the mentioned problems. The computational results show that the new method proposed in this paper is very efficient, robust and stable in practical computation. First of all, the new method in the paper avoids the evaluation of second derivatives of f. Secondly, the storage of any matrix associated with the Newton type method is avoided at each iteration.

Moreover, the new method needs less iterative number and less evaluation number of f and then less total CPU time than FR, PRP, HS, and steepest descent method. The memory gradient method with trust region is more suitable to solve large scale unconstrained optimization problems. In fact, we compare the results of new method with that of restart conjugate gradient method and other conjugate gradient method without line searches. The comparison shows that the search direction of the new algorithm is a good search direction at each iteration.

7. Conclusion

In this paper we presented a new memory gradient method with trust region for unconstrained optimization problems. The method combines line search method and trust region method to generate new iterative points at each iteration and therefore has both advantages of line search methods and trust region methods. It sufficiently uses previous multi-step iterative information at each iteration and avoids the storage and computation of matrices associated with Hessian of objective functions, so that it is suitable to solve large scale optimization problems. The paper also designed an implementable version of the new method and analyzed its global convergence under weak conditions. This idea makes us design some quick convergent, effective, and robust algorithms since it uses more information from previous iterative steps and has a trust region. Preliminary numerical experiment shows that the new method is effective and robust in practical computation, compared with other similar methods.

For the future research, we may choose search direction at each iteration as

$$d_k = -\beta_k^{(k)} g_k - \sum_{i=2}^{m} \beta_{k-i+1}^{(k)} d_{k-i+1}, k \geq m$$

or

$$d_k = -\beta_k^{(k)} g_k - \sum_{i=2}^{m} \beta_{k-i+1}^{(k)} (x_k - x_{k-i+1}), k \geq m$$

or other forms of linear combination of previous iterative information. Moreover, we can use the trust region approach to design memory gradient algorithms without line search.

Acknowledgements

The authors would like to thank the referees and the editor for their valuable comments and suggestions that greatly improved the paper.

References

[1] J. Barzilai and J.M. Borwein, Two-point step size gradient methods, IMA J. Numer. Anal. 8 (1988) 141–148.
[2] D.P. Bertsekas, *Constrained Optimization and Lagrange Multiplier Methods* (Academic, New York, 1982).
[3] J.W. Cantrell, Relation between the memory gradient method and the Fletcher–Reeves method, J. Optim. Theory Appl. 4 (1969) 67–71.
[4] E.E. Cragg and A.V. Levy, Study on a supermemory gradient method for the minimization of functions, J. Optim. Theory Appl. 4 (1969) 191–205.
[5] A.R. Conn, N. Gould, A. Startenaer and P.L. Toint, Global convergence of a class of trust region algorithms for optimization using inexact projections on convex constraints, SIAM J. Optim. 3 (1993) 164–221.
[6] L.W.C. Dixon, Conjugate directions without line searches, J. Inst. Math. Appl. 11 (1973) 317–328.
[7] Y.H. Dai and Y. Yuan, *Nonlinear Conjugate Gradient Method* (Shanghai Science and Technology, 1999).

[8] A. Friedlander, J.M. Martinez, B. Molina and M. Raydan, Gradient method with retards and generalizations, SIAM J. Numer. Anal. 36 (1999) 275–289.

[9] L. Grippo, F. Lampariello and S. Lucidi, A class of nonmonotone stability methods in unconstrained optimization, Numer. Math. 62 (1991) 779–805.

[10] J.C. Gilbert and J. Nocedal, Global convergence properties of conjugate gradient methods for optimization, SIAM J. Optim. 2 (1992) 21–42.

[11] L. Grippo and S. Lucidi, A globally convergent version of the Polak–Ribière conjugate gradient method, Math. Program. 78 (1997) 375–391.

[12] H.Y. Huang and J.P. Chambliss, Quadratically convergent algorithms and one-dimensional search schemes, J. Optim. Theory Appl. 11 (1973) 175–188.

[13] M.R. Hestenes, *Conjugate Direction Methods in Optimization* (Springer Berlin Heidelberg New York, 1980).

[14] R. Horst, P.M. Pardalos and A.V. Thoai, *Introduction to Global Optimization* (Kluwer, Dordrecht, 1995).

[15] D.C. Luenberger, *Linear and Nonlinear Programming*, second edition (Addition Wesley, Reading, MA, 1989).

[16] A. Miele and J.W. Cantrell, Study on a memory gradient method for the minimization of functions, J. Optim. Theory Appl. 3 (1969) 459–470.

[17] J. Nocedal and S.J. Wright, *Numerical Optimization* (Springer, Berlin Heidelberg New York, 1999).

[18] J.M. Ortega and W.C. Rheinboldt, *Iterative Solution of Nonlinear Equations in Several Variables* (Academic Press, New York, 1970).

[19] M. Raydan, The Barzilai and Borwein gradient method for large scale unconstrained minimization problems, SIAM J. Optim. 7 (1997) 26–33.

[20]. M. Raydan and B.F. Svaiter, Relaxed steepest descent and Cauchy–Barzilai–Borwein method, Comp. Optim. Appl. 21 (2002) 155–167.

[21] J.A. Snyman, A new and dynamic method for unconstrained minimization, Appl. Math. Model. 6 (1982) 449–462.

[22] J. Schropp, A note on minimization problems and multistep methods, Numer. Math. 78 (1997) 87–101.

[23] J. Schropp, One-step and multistep procedures for constrained minimization problems, IMA J. Numer. Anal. 20 (2000) 135–152.

[24] Z.J. Shi, Modified HS conjugate gradient method and its global convergence (in Chinese), Math. Numer. Sin. 23 (2001) 333–406.

[25] Z.J. Shi, Restricted PR conjugate gradient method and its convergence (in Chinese), Adv. in Math. 31 (2002) 47–55.

[26] Z.J. Shi and J. Shen, A gradient-related algorithm with inexact line searches, J. Comput. Appl. Math. 170 (2004) 349–370.

[27] A. Vardi, A trust region algorithm for equality constrained minimization: Convergence properties and implementation, SIAM J. Numer. Anal. 22 (1985) 575–591.

[28] D.J. Van Wyk, Differential optimization techniques, Appl. Math. Model. 8 (1984) 419–424.

[29] K. Schittkowski, *More Test Examples for Nonlinear Programming Codes, Lecture Notes in Economics and Mathematical Systems*, 282 (Springer, 1987).

[30] M.N. Vrahatis, G.S. Androulakis, J.N. Lambrinos and G.D. Magolas, A class of gradient unconstrained minimization algorithms with adaptive stepsize, J. Comput. Appl. Math. 114 (2000) 367–386.

[31] M.A. Wolfe and C. Viazminsky, Supermemory descent methods for unconstrained minimization, J. Optim. Theory Appl. 18 (1976) 455–468.